(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **61127044 A**

(43) Date of publication of application: **14.06.86**

(51) Int. Cl **G06F 11/28**
G06F 9/44

(21) Application number: **59250174**

(22) Date of filing: **27.11.84**

(71) Applicant: **FUJITSU LTD**

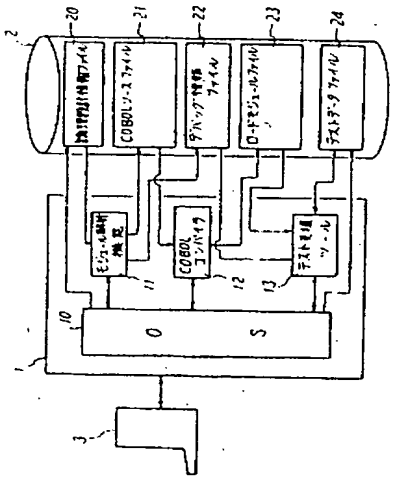
(72) Inventor: **YANAGISAWA MINORU**

(54) TEST SYSTEM FOR SINGLE UNIT PROGRAM

(57) Abstract:

PURPOSE: To improve the developing efficiency of a program by providing a memory to a test back-up tool which replaces modules to start a module to be tested after the test data is set or setting and delivering the module during its test.

CONSTITUTION: A test back-up tool 13 is provided to a processor 1 connected to a console 3 in addition to a module analysis function 11 and COBOL compiler 12. An operating system 10 controls these function 11, compiler 12 and tool 13 respectively. A test data file 24 is provided to a file memory 2 and then connected to the tool 13. The test data on the module to be tested is stored in the file 24 and the tool 13 is started. A parameter delivery area is secured in an internal memory of the processor 1 if a high-order module exists. Then the data is set to start a test. While said delvery area is secured in the internal memory when no low-order load module exists. Then the data is set through the file 24 and used as the output data of low-order module.



⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　昭61－127044

⑤1Int.Cl.⁴	識別記号	庁内整理番号	④3公開　昭和61年(1986)6月14日
G 06 F 11/28		7343－5B	
9/44		Z－8120－5B	

審査請求　未請求　発明の数 1　(全6頁)

⑤4発明の名称　プログラムの単体テスト方式

②1特　願　昭59－250174

②2出　願　昭59(1984)11月27日

⑦2発明者　柳沢　実　川崎市中原区上小田中1015番地　富士通株式会社内

⑦1出願人　富士通株式会社　川崎市中原区上小田中1015番地

⑦4代理人　弁理士　山谷　晧榮

明　細　書

1. 発明の名称　プログラムの単体テスト方式

2. 特許請求の範囲

複数のモジュールが階層的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において，該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け，該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には，該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ，該被テストモジュールが該下位モジュールを呼出す場合には，該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴とするプログラムの単体テスト方式。

3. 発明の詳細な説明

〔産業上の利用分野〕

本発明は，複数のモジュールで構成されるプログラムの各モジュールプログラムを単体でテストするプログラムの単体テスト方式に関し，特に関連する他のモジュールが作成されていなくてもテストできるプログラムの単体テスト方式に関する。

一般に大きなシステムにおいては，プログラムが1本のモジュール(コンパイル単位)で構成されているものは少なく複数のモジュールが集まって一つの処理を行なうように階層的に構成されている。

例えば，第5図に示すようにモジュールAのサブルーチンとしてモジュールBが，モジュールBのサブルーチンとしてモジュールC，Dが構成されて1つのプログラムを構成するようにしている。このような階層的なプログラムを開発するには，テストの効率を考慮してモジュールの作成順序等が決定され，一般に第6図で示す如くシステム設

計された後，各モジュールA，B，C，Dが作成される。

この開発の手順としては，上位のモジュールAから作成するトップダウン方式と，下位のモジュールC又はDから作成するボトムアップ方式とがある。

〔従来の技術〕

このような各モジュールが階層的に構成されたプログラムをテストするには，全てのモジュールが完成してからプログラムをテストすることが望ましいが，各モジュールが順次作成されていくような場合には，各モジュールの作成毎にそのモジュールを単体でテストするほうが開発手順上効率がよい。

しかしながら，モジュールを単体でテストするにしても，階層的なプログラムの構成では，関連する他のモジュールが存在しないとテストできない。例えばモジュールAをテストするときは，モジュールBが必要であり，モジュールBをテストするときは，モジュールA，C，Dが必要となる。

このため，従来は，テストするモジュールに関連する仮モジュールをテスト担当者が作成し，係るモジュールの単体テストを行っていた。

例えば，モジュールBをテストするため上位のモジュールAの仮モジュール（ドライバという）及び下位のモジュールC，Dの仮モジュール（スタブという）を作成していた。

〔発明が解決しようとする問題点〕

このようなモジュールのテストで必要となるドライバやスタブは仮モジュールなので小さく簡単に作成できるが，通常のプログラム作成と同様にコーディングやこれのテストの工数を要し，単体テストのためのドライバやスタブの開発をモジュール個々に要するという問題があった。

〔問題点を解決するための手段〕

本発明は，テストされるモジュールの必要とするドライバ又はスタブを自動的に発生させ，仮モジュールの作成を必要としないで単体テストできるプログラムのテスト方式を提供するにある。

このため，本発明は，複数のモジュールが階層的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において，該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け，該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には，該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ，該被テストモジュールが該下位モジュールを呼出す場合には，該被テストモジュールのテスト実行中に該下位モジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴としている。

〔作用〕

本発明では，テスト支援ツール（プログラム）が設けられるが，この支援ツールがドライバ又はスタブ機能を自動的に実行するようにしている。

即ち，支援ツールは被テストモジュールを解析し，これに含まれる上位又は下位モジュールのインターフェイス（接続）情報を抽出し，上位又は下位モジュールの存在を検出し，上位モジュールが存在する時は被テストモジュールの起動に先立ち，インターフェイス情報を基にパラメータ領域をメモリに確保し，更にこの領域に外部から与えられたテストデータをセットして被テストモジュールのテスト起動を行ない，又下位モジュールが存在するときは，該被テストモジュールのテスト実行中に下位モジュールを呼出すことに応じてインターフェイス情報から外部から与えられたテストデータをメモリにセットして被テストモジュールに渡すようにして被テストモジュールのテストを続行させるようにしている。

〔実施例〕

以下，本発明を実施例により詳細に説明する。

第1図は本発明の一実施例ブロック図であり，図中，1はコンピュータ（プロセッサ）であり，プログラムを実行して所望の処理を行なうもの，

１０はそのＯＳ（オペレーティングシステム）部であり，コンピュータ１全体の制御を行なうもの，１１はモジュール解析部であり，後述する如く論理設計言語で記述された論理設計情報を解析してＣＯＢＯＬのソースプログラムを作成し，且つインターフェイス（デバック）情報を抽出するもの，１２はＣＯＢＯＬコンパイラであり，ソースプログラムをコンピュータの実行できる形式のロードモジュールに変換するもの，１３はテスト支援ツール部であり，後述する如くインターフェイス情報からドライバ／スタブ機能を行なうものである。尚，ＯＳ部１０，モジュール解析部１１，ＣＯＢＯＬコンパイラ１２及びテスト支援ツール部１３はコンピュータ１の実行する機能をプログラム化したものである。２はファイルメモリであり，各種ファイルを格納しておくもの，２０は論理設計情報ファイルであり，プログラマーが作成した後述する論理設計情報を格納しておくもの，２１はＣＯＢＯＬソースファイルであり，モジュール解析部１１によって作成されたＣＯＢＯＬソースプログラムを格納しておくもの，２２はデバック情報ファイルであり，モジュール解析部１１で抽出されたインターフェイス情報を格納しておくもの，２３はロードモジュールファイルであり，ＣＯＢＯＬコンパイラ１２が作成したロードモジュールを格納しておくもの，２４はテストデータファイルであり，オペレータが入力したテストデータを格納しておくもの，３はコンソールであり，ディスプレイとキーボードから成り，オペレータが操作して論理設計情報やテストデータを入力するものである。

次に，オ１図実施例構成の動作について，オ２図の処理フロー図，オ３図の論理設計情報説明図及びオ４図のテスト支援ツールの動作説明図を用いて説明する。

以下，周知の高級言語である論理設計言語によりプログラミングされたものについてその動作を説明する。

① 論理設計言語によるプログラミングは，オ３図に示す如く各モジュールＡ，Ｂ，Ｃ，Ｄで共通なフォーマット定義書（オ３図(A)），各モジュールのインターフェイス定義書（オ３図(B)），各モジュールのモジュール定義書（オ３図(C)）等を作成することにより行なう。この共通フォーマット定義書は各モジュールで共通な項目（データ）の形式，桁数等の定義を行なうものであり，インターフェイス定義書は各モジュールの入出力項目，モジュール名等を定義するためのものであり，基本設計段階で決定され，コンソール３よりコンピュータ１を介しファイルメモリ２の論理設計情報ファイルに格納されている。

一方，モジュール定義書は各モジュール毎に作成され，当該モジュールの処理や論理，その条件が手順に従って羅列されたものであり，プログラマーは各モジュールに必要な処理等を順次記述してプログラミングし，これが当該モジュールの処理内容を示す。オペレータはこのモジュール定義書を完成する毎にコンソール３からコンピュータ１を介し論理設計情報ファイル２０に格納せしめる。

例えば，モジュールＢのモジュール定義書が完成し，論理設計情報ファイル２０に格納され，他のモジュールＡ，Ｃ，Ｄは完成していないとする。

② オペレータはコンソールよりコンピュータ１にＯＳ部１０のジョブ制御機能を働かせ，モジュールＢのＣＯＢＯＬソースプログラムを作成するように指令する。これによってコンパイラの一種であるモジュール解析部１１は論理設計情報ファイル２０の論理設計情報を読出し，ＣＯＢＯＬで記述されたソースプログラムを作成し，ＣＯＢＯＬソースファイル２１に格納する。このＣＯＢＯＬで記述されたソースプログラムは，例えばオ４図の被テストモジュールとして示すものであり，論理設計言語を用いずプログラマーがＣＯＢＯＬでプログラミングしたものに相当する。

③ これとともにモジュール解析部１１は論理設計情報からインターフェイス情報を抽出する。例えば，オ３図の例では，モジュールＢはモジュールＡに呼び出され，パラメータの数はＧ１，Ｇ２の２ツでその領域はＡ１，Ａ２であることがモジ

ユールBのインターフェイス定義書より抽出され，そのパラメータG1，G2の長さは共通フォーマット定義書より抽出される。又，モジュールBのモジュール定義書よりモジュールC（日数＝%日数計算），モジュールDを呼び出すことが抽出され，その使用領域はモジュールC，Dのインターフェイス定義書より抽出される。

これをCOBOLソースプログラムの記述から見ると，第4図の被テストモジュールの記述の如く，モジュールAのA1，A2を用い，モジュールCのB1，モジュールDのB2を用いることになる。

モジュール解析部11はモジュールBが呼び出されるときのインターフェイス（モジュール名，パラメータの数，パラメータの領域名，その長さ等）及びモジュールBが呼び出す時のインターフェイスを論理設計情報から抽出し，デバッグ情報ファイル22に格納する。

このようにしてモジュールBに対するCOBOLソースプログラムが作成され，且つモジュールトモジュールBに渡すパラメータの受け渡し領域をコンピュータ1の図示しない内部メモリに確保する。前述の例では，第4図に示す如く内部メモリにA1及びA2の入出力領域を確保し，そのアドレスをセットしておく。

次に，コンソール3よりオペレータにテストデータの指示を表示し，オペレータはテストデータファイル24のどのテストデータをセットするかをコンソール3より指示する。これによつて内部メモリのA1の領域にテストデータファイル24の必要なテストデータ（上位モジュールからの入力データ）がセットされることになる。

これによつてモジュールBはパラメータA1を用いて処理を実行できる。尚，A2はモジュールBの実行結果の出力のためのものである。

⑥　次に，テスト支援ツール部13はOS部10に対し被テストモジュールの起動を指示する。

OS部10はロードモジュールファイル23のモジュールBのロードモジュールを実行し，その中でモジュールAのパラメータA1を用いて行なう。

⑦　このようにしてモジュールBのテストを実行していくうちに，モジュールC又はDの"call"（呼び出し）命令があり，OS部10がロードモジュールファイル23を探索した結果モジュールC，Dのロードモジュールが見付からなかつたとすると，エラーをテスト支援ツール部13に通知する。

⑧　これによつてテスト支援ツール部13は，被テストモジュールが下位モジュールを呼ぶために用意した内部メモリのパラメータ領域（B1やB2）のアドレスを得，コンソール3よりオペレータにテストデータの指示を表示する。オペレータはテストデータファイル24のどのテストデータをセットするかをコンソール3より指示する。これによつて内部メモリのパラメータ領域にテストデータファイル24の必要なテストデータ（下位モジュールの出力データ）がセットされることになる。

従つて被テストモジュールBはこのテストデー

間のインターフェイス情報が抽出される。

④　次に，モジュールBの単体テストを行なう。

このため，オペレータはコンソール3よりコンピュータ1を介しモジュールBのテストのためのテストデータをテストデータファイル24に格納する。そして，オペレータはテスト実行指令をコンピュータ1に入力する。コンピュータ1では，OS部10がCOBOLコンパイラ12を動作せしめ，COBOLソースファイル21のモジュールBのCOBOLソースプログラムをコンピュータ1の実行できるロードモジュールに翻訳してロードモジュールファイル23に格納する。次にOS部10はテスト支援ツール部13を起動し，デバッグ情報ファイル22のインターフェイス情報を解析せしめる。

⑤　テスト支援ツール部13は，第2図の処理フロー図の如く前述のインターフェイス情報を解析し，上位モジュールが存在するかを検出する。上位モジュールが存在するとドライバ機能，即ち上位モジュール代替機能を実行する。先づ，被テス

タを下位モジュールの出力データとして受け取り，更にテストを続行していく。

このようにして，テスト支援ツール部13が，インターフェイス情報から上位又は下位モジュールの代替機能を自動的に実行し，被テストモジュールのテストを行なわしめる。

上述の実施例では，論理設計言語でプログラムした例について説明したが，COBOLによってプログラムしてソースプログラムを作成してもよく，この場合COBOLソースプログラムからインターフェイス情報が抽出される。又，テストデータをパラメータ領域にセットするのにコンソール3よりオペレータが指示しているが，これを自動的に行なってもよい。更にソースプログラムの言語もCOBOLに限らず，他の言語を用いてもよい。

以上本発明を一実施例により説明したが，本発明は本発明の主旨に従い種々の変形が可能であり，本発明からこれらを排除するものではない。

〔発明の効果〕

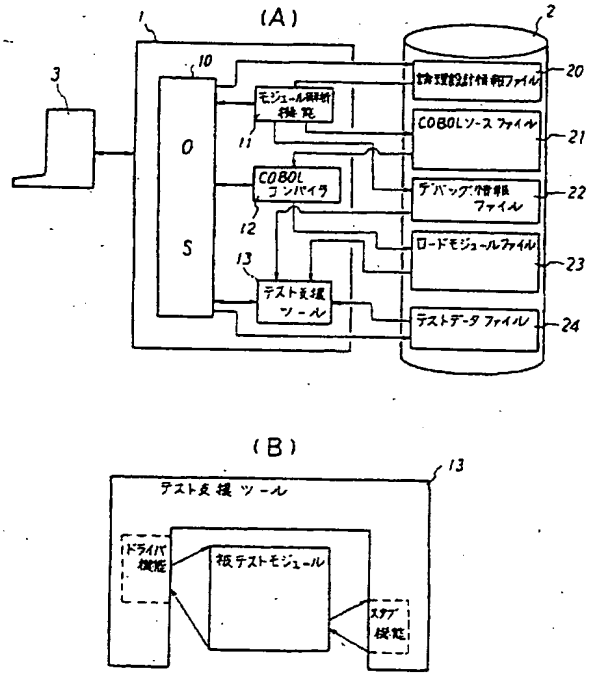
以上説明した様に，本発明によれば，複数のモジュールが階層的に構成されるプログラムのモジュール単体をプロセッサが実行してテストを行うプログラムの単体テスト方式において，該被テストモジュールを解析して該被テストモジュールに対する上位又は下位のモジュールの代替機能を行うテスト支援ツールを設け，該テスト支援ツールは該被テストモジュールが該上位モジュールから呼出される場合には，該被テストモジュールに渡すパラメータの領域をメモリに確保してテストデータをセット後該被テストモジュールを起動せしめ，該被テストモジュールが該下位モジュールを呼出す場合には，該被テストモジュールのテスト実行中に該下位もジュールが呼出されたことに応じて被テストモジュールに渡すテストデータを該メモリにセットして該被テストモジュールに渡すことを特徴としているので，関連する他のモジュールが完成していなくても被テストモジュールのテストを自動的に実行できるという効果を奏し，係るモジュールが階層的に構成されたプログラムの開発効率を高めることができる。又，被テストモジュールに対するドライバ，スタブをプログラミングしなくてよいため，ドライバやスタブのテストのための作業をしなくて済むという効果も奏し，一層プログラム作成を容易に且つ短期間で完成せしめることに寄与するところが大きい。

4. 図面の簡単な説明

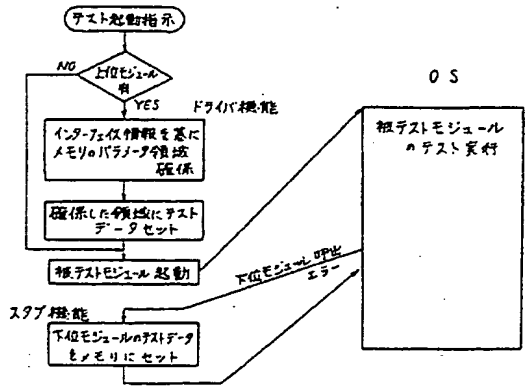
第1図は本発明の一実施例ブロック図，第2図は第1図実施例構成のテスト支援ツールの処理フロー図，第3図は第1図実施例構成に用いられる論理設計言語の説明図，第4図は第1図実施例構成のテスト支援ツールの動作説明図，第5図は階層的モジュール構成プログラムの説明図，第6図は第5図構成のプログラムの開発手順説明図である。

図中，1…コンピュータ（プロセッサ），2…ファイルメモリ，3…コンソール，13…テスト支援ツール。

第1図



第 2 図



第 3 図

(A)

共通フォーマット定義

項番	区分	項目	形式	条件
10		口座番号	9(4)	
20		科目コード	9(4)	
30		氏名	N(6)	

(B)

インターフェイス定義 (モジュールB)

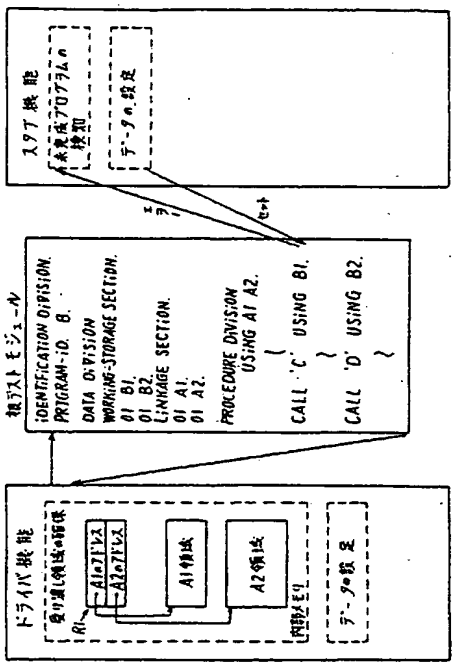
項番	区分	エントリモジュール	入出力項目
1	S	マル停管理B	GRP:G1=A1 G2=A2

(C)

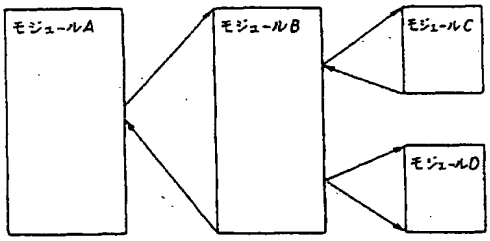
モジュール定義 (モジュールB)

項番	区分	処理	論理	条件
10				
20				
30		日数の算出		

第 4 図



第 5 図



第 6 図